

An Efficient Parameter Space Search as an Alternative to Markov Chain Monte Carlo

Scott F. Daniel¹, Andrew J. Connolly¹, and Jeff Schneider²

¹*Department of Astronomy, University of Washington, Seattle, WA*

²*Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA*

3 March 2013

ABSTRACT

We consider the problem of inferring constraints on a high-dimensional parameter space with a computationally expensive likelihood function. Markov chain Monte Carlo (MCMC) methods offer significant improvements in efficiency over grid-based searches and are easy to implement in a wide range of cases. However, MCMCs offer few guarantees that all of the interesting regions of parameter space are explored. We propose a machine learning algorithm that improves upon the performance of MCMC by intelligently targeting likelihood evaluations so as to quickly and accurately characterize the likelihood surface in both low- and high-likelihood regions. We compare our algorithm to MCMC on toy examples and the 7-year WMAP cosmic microwave background data release. Our algorithm finds comparable parameter constraints to MCMC in fewer calls to the likelihood function and with greater certainty that all of the interesting regions of parameter space have been explored.

1 INTRODUCTION

Researchers in many fields use Markov chain Monte Carlo (MCMC) methods to translate data into inferences on high-dimensional parameter spaces (Gilks *et al.* 1996). Studies of the cosmic microwave background (CMB) anisotropy spectra are a perfect example of this practice (Lewis and Bridle 2002; Dunkley *et al.* 2005). The concordance cosmology requires six or seven (depending on whether or not one assumes spatial flatness) parameters to theoretically describe the spectrum of CMB anisotropies. Given the computational expense of converting just one point in this parameter space into an anisotropy spectrum (1.3 seconds using the Boltzmann code CAMB (Lewis, Challinor, and Lasenby 2000) on a 2.5 GHz dual-core machine) and then comparing that spectrum to the data (2 seconds using the official WMAP likelihood code (WMAP 2010)), exhaustively exploring the parameter space in search of models that fit the data well is unfeasible. MCMC is an alternative to this costly process that randomly walks through parameter space such that exploration theoretically locates and confines itself to regions where the fit to the data is good, and thus integrates the distribution of parameter values only over that space where it has high support. Integrations that might have taken months when done exhaustively can be accomplished in days; those that might have taken days can be accomplished in hours. It is fair to

say that MCMC has become an “industry standard” within some communities. In spite of this success, serious questions remain. MCMC is a sampling method and its theoretical guarantees are mostly with respect to convergence rather than sampling efficiency (Atchadé and Rosenthal 2005). Parameter fitting problems are search problems, not sampling problems. There is no fundamental reason that a sampling algorithm should be good for searching and certainly no guarantee that the performance of MCMC as a search algorithm with a finite number of samples will be particularly efficient. Finally, MCMC methods generally force investigators to make Bayesian assumptions which may or may not be desirable.

We present an algorithm originally proposed and implemented by (Bryan *et al.* 2007b) as a more efficient and flexible alternative to MCMC. We will refer to the algorithm as the Active Parameter Search (APS) algorithm. We provide C++ code to implement APS (publicly available at <https://github.com/uwssg/APS>) and test it on the 7-year release of the WMAP CMB data (Jarosik *et al.* 2011). Section 2 presents an overview of both the MCMC and APS algorithms and discusses the shortcomings of the former and how the latter attempts to address them. Section 3 details the user-specified parameters necessary to APS and some of the attendant considerations. Section 4 compares the performance of MCMC and APS on a toy model of a multi-

modal likelihood function. Section 5 presents the results of the WMAP 7 test. We find that APS achieves parameter constraints at comparable (if not faster) times to MCMC while simultaneously exploring the parameter space more broadly.

2 MCMC

We begin this section with a general discussion of the process of deriving parameter constraints from data. We then sketch an MCMC method called the Metropolis-Hastings algorithm (Gilks *et al.* 1996; Lewis and Bridle 2002). We will outline the APS algorithm with an aside discussing Gaussian processes, a method of multi-dimensional function fitting which occupies some prominence in the algorithm. We end this section by directly comparing APS with MCMC on a toy model in a two-dimensional parameter space.

The basic problem both MCMC and APS attempt to address is the following: suppose there is a theory described by some number of parameters $\vec{\theta} = \{\theta_1, \theta_2, \theta_3, \dots\}$. This theory makes a prediction about some function f which is tested by an experiment resulting in N_D data points $\{d_1, d_2, \dots, d_{N_D}\}$. To quantify what these data say about the theory, an investigator wants to ask what combinations of values of $\vec{\theta}$ result in predictions that are consistent with the data to some threshold probability $(100 - \alpha)\%$. Usually, this is quantified by assuming that the N_D data points are independent and normally distributed so that the statistic

$$\chi^2 \equiv \sum_i^{N_D} \left(\frac{f_i - d_i}{\sigma_i} \right)^2$$

(where σ_i is the uncertainty associated with the d_i data-point) is a random variable distributed according to the χ^2 probability distribution \mathcal{P}_{χ^2} with N_D degrees of freedom. In that case, the $(100 - \alpha)\%$ confidence limit constraint on $\vec{\theta}$ corresponds to the set of all values $\vec{\theta}$ which result in values of χ^2 such that

$$\int_0^{\chi^2} d\chi'^2 \mathcal{P}_{\chi^2}(\chi'^2) \leq (100 - \alpha)\%$$

More generally, one has a likelihood function \mathcal{L} (above represented by \mathcal{P}_{χ^2}) quantifying the goodness-of-fit between data and theory. One uses MCMC or APS to find the values of $\vec{\theta}$ corresponding to $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$, where \mathcal{L}_{lim} is some threshold set by statistics and the desired $(100 - \alpha)\%$.

Bayesian inference approaches this problem indirectly by assuming that $\vec{\theta}$ are random variables distributed according to the distribution function \mathcal{L} . Constraining these parameters to $(100 - \alpha)\%$ is therefore a problem of integrating \mathcal{L} over $\vec{\theta}$ until the integral contains $(100 - \alpha)\%$ of the total \mathcal{L} . MCMC assumes this approach, as will be shown below. A common criticism of this mode of thought is that, even if the theory is a poor description of the data, Bayesian inference

will still yield a constraint consisting of the $(100 - \alpha)\%$ least-poor region of parameter space. Frequentist inference takes a more objective approach, defining the $(100 - \alpha)\%$ confidence limit to be all points in parameter space that satisfy $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$. APS assumes this approach, as will also be shown below, and exploits it to derive parameter constraints with significantly fewer calculations of \mathcal{L} (a presumably costly procedure) than MCMC.

2.1 Sketch of the MCMC algorithm

The direct product of MCMC is a chain: a list of points in parameter space that MCMC sampled and the number of times it sampled them.

- (1M) Begin by sampling a point $\vec{\theta}_1$ from parameter space at random. Record this point in the chain and evaluate the likelihood function for that combination of parameters. The value of this likelihood call will be \mathcal{L}_1 .

- (2M) Select another point $\vec{\theta}_2$ in parameter space. This point should be selected randomly, but according to some distribution that depends only on the step length $|\vec{\theta}_1 - \vec{\theta}_2|$ and is tuned so that the algorithm explores away from the initial point but does not effectively jump to any other place in parameter space uniformly at random. Evaluate the likelihood function at $\vec{\theta}_2$, giving you \mathcal{L}_2 .

- (3M) If $\mathcal{L}_2 > \mathcal{L}_1$ (i.e., if the new point in parameter space is a better fit to the data than the old point), the step is accepted and the new point is recorded in the chain. Set $\mathcal{L}_1 = \mathcal{L}_2$ and $\vec{\theta}_1 = \vec{\theta}_2$.

- (4M) If $\mathcal{L}_2 < \mathcal{L}_1$, draw a random number β between 0 and 1. If $\beta < \mathcal{L}_2/\mathcal{L}_1$, accept the step anyway.

- (5M) If the step was ultimately rejected, increment the number of samples associated with $\vec{\theta}_1$ by one.

- (6M) Return to step (2M) and repeat. There are heuristic statistics that can be performed on the chain to give an indication of whether it has adequately sampled parameter space (Brooks and Gelman 1998), though they have issues we point out later.

This is the most basic form an MCMC algorithm can take. In addition to Chapter 1 of (Gilks *et al.* 1996), one may wish to consult appendix A of (Lewis and Bridle 2002) for guidance writing a functional MCMC.

The tests in step (3M) and (4M) mean that any MCMC run will gravitate towards the high-likelihood regions of parameter space and ignore regions of extremely low likelihood. One converts the chain into parameter inferences by gridding parameter space and treating the chain as a histogram of how many times the chain visited each point on the grid (or its proximity). This histogram will closely

match the Bayesian posterior distribution over the parameters. $(100 - \alpha)\%$ confidence limits are derived by integrating this histogram from its peak out to whatever bounds contain $(100 - \alpha)\%$ of the sampled points.

There are several shortcomings with this method that the APS addresses. The first shortcoming is efficiency. Because MCMC seeks to take samples that match the distribution rather than learning specific information about the distribution, such as where the confidence boundaries are, it wastes samples in areas where it already has information. For example, once a high-likelihood parameter setting has been evaluated there is no need to evaluate there again. It is already known that this is a high-likelihood region of the space. However, MCMC specifically indicates that high-likelihood by putting more samples there. These are wasted evaluations.

A second MCMC shortcoming, related to the first, involves the possibility that there are several disjoint regions of high likelihood in the parameter space under consideration. This can be problematic for MCMC because steps (2M)-(4M) ensure that, once the chain finds one of these high likelihood regions, it is unlikely to step out of it and find another (unless the distribution in step (2M) is such that it allows very large steps, in which case the chain will take significantly longer to converge). An unexplored region of the space may remain unexplored for a long time (though in theory not infinitely long). MCMC never determines whether it has explored the region and whether information could be learned by doing so.

When initially testing APS, Bryan *et al.* found a separate region of high-likelihood parameter space in the 1-year WMAP data release which had been totally ignored by MCMC analyses. This second peak in \mathcal{L} disappeared as the WMAP signal-to-noise improved in subsequent data releases (as we will see in Section 5), but it does serve as an illustration of this particular peril of MCMC. Dunkley *et al.* (2005) and (Atchadé and Rosenthal 2005) attempt to address this problem and the general inefficiency of MCMC as a search algorithm by optimizing the proposal density in step (2M). They find no clear solution that guards against multiple high likelihood regions.

The final shortcoming of MCMC involves its unequivocally Bayesian nature. For a number of data points $N_D \gg 1$ Bayesian and frequentist confidence limits are approximately equivalent. They can, however, yield conflicting results when constraining a subset of parameter space. Consider a six-dimensional parameter space like the space of cosmological parameters mentioned in Section 1 and revisited in Section 5. If one is really only interested in constraining $\{\theta_1, \theta_2\}$, one still has to deal with the question of how to treat $\{\theta_3, \dots, \theta_6\}$, since they presumably affect \mathcal{L} in a way that is non-trivially correlated with $\{\theta_1, \theta_2\}$. In Bayesian inference, one deals with this question by integrating the probability distribution over the full range of the uninteresting parameters (marginalizing over them), i.e. the $(100 - \alpha)\%$

confidence limit is the contour in $\{\theta_1, \theta_2\}$ space such that

$$(100 - \alpha)\% \times \int_{-\infty}^{\infty} d^6 \theta \mathcal{L}(\vec{\theta}) = \int_{\theta_{1i}}^{\theta_{1f}} d\theta_1 \int_{\theta_{2i}}^{\theta_{2f}} d\theta_2 \int_{-\infty}^{\infty} d\theta_3 \int_{-\infty}^{\infty} d\theta_4 \int_{-\infty}^{\infty} d\theta_5 \int_{-\infty}^{\infty} d\theta_6 \mathcal{L}(\vec{\theta}) \quad (1)$$

where $\{\theta_{1i}, \theta_{1f}\}, \{\theta_{2i}, \theta_{2f}\}$ represent the bounds of the confidence limit in the two-dimensional subspace of interest. This integration yields the two-dimensional ellipses of figures 4 and 6 of (Lewis and Bridle 2002) (and most observational cosmology papers since). From a frequentist perspective, the integration in equation (1) is problematic. It risks drawing a bound that is either too restrictive or too inclusive. An over-restrictive bound could arise because the integral (1) weights points in $\{\theta_1, \theta_2\}$ space according to the density of corresponding points in $\{\theta_1, \dots, \theta_6\}$ space that are highly likely. If there are combinations of $\{\theta_1, \theta_2\}$ that are highly likely only for an extremely limited set of $\{\theta_3, \dots, \theta_6\}$, those points will be excluded from the $(100 - \alpha)\%$ confidence limit even though they are strictly allowed according to the $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$ criterion. Bayesian inference does not give much thought to these excluded points because, if we consider the parameters $\vec{\theta}$ to be random (which Bayesians do), the excluded points correspond to highly improbable values of $\{\theta_1, \theta_2\}$ precisely because they only agree with the data for such a limited set $\{\theta_3, \dots, \theta_6\}$. Frequentist bounds, consisting of any and all points for which $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$, do not care how many different points in the larger parameter space are acceptable for a given $\{\theta_1, \theta_2\}$. If just one point satisfies $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$, the corresponding point in subspace is within the bound.

An over-inclusive bound could arise because the integral (1) searches only for that bound which contains $(100 - \alpha)\%$ of the total probability. If the model chosen to describe the data is universally poor, MCMC will still return a $(100 - \alpha)\%$ consisting of the least-poor region in parameter space without any obvious warning that the model is probably wrong. A frequentist bound would, in that case, return no $(100 - \alpha)\%$ confidence bound, highlighting the model's deficiency.

Traditionally, Bayesian assumptions are made because they do not require prior knowledge of the underlying likelihood function. In order to draw a Frequentist confidence limit, one must be able to calculate \mathcal{L}_{lim} before sampling any points in parameter space. Bayesian confidence limits only require knowledge of the relative likelihood between sampled points and thus can be revised with each new sample. This represents one advantage MCMC retains over APS.

2.2 Active Parameter Search

To address the shortcomings of MCMC, Bryan *et al.* Bryan *et al.* (2007b) propose the following alternative algorithm for exploring parameter space.

- (1A) Generate some number N_S of starting points

uniformly distributed across parameter space. Evaluate \mathcal{L} at each of these points. Store them in $\{\vec{\theta}_i, \mathcal{L}_i\}$.

- (2A) Use the points already evaluated to fit a surrogate function that estimates the likelihood and uncertainty in it for other, as yet unevaluated, candidate points.

- (3A) Generate a large number N_C of candidate points also uniformly distributed over parameter space. Use the surrogate function to guess the value of \mathcal{L} at each of the candidate points. This guess will be μ . The uncertainty in your guess will be σ . Section 2.3 will describe one means of finding μ and σ . The assumption is that estimating μ and σ is orders of magnitude less computationally expensive than finding the true \mathcal{L} .

- (4A) Select the candidate point with the maximum value of the statistic

$$S \equiv c\sigma - |\mathcal{L}_{\text{lim}} - \mu| \quad (2)$$

where c is a parameter that has a similar effect as the length parameter in MCMC's proposal distribution. Small values will make it take more samples around already good (high \mathcal{L}) points while large values will make it explore more aggressively. Evaluate \mathcal{L} at the selected point and add it to the list of evaluated $\{\vec{\theta}_i, \mathcal{L}_i\}$.

- (5A) Repeat steps (2A) through (4A). Confidence bounds are found by gridding the parameter space (though no integration or other accumulation is required). Convergence can be estimated heuristically by observing the size of changes in confidence bounds.

Maximizing S in equation (2) implies, to some extent, minimizing $|\mu - \mathcal{L}_{\text{lim}}|$. This algorithm therefore seeks out the boundary of the $(100 - \alpha)\%$ confidence limit, rather than its interior. This yields some efficiency improvements over MCMC.

The dependence of S on σ – the uncertainty of the predicted candidate \mathcal{L} value – forces the algorithm to simultaneously pay attention to unexplored regions of parameter space. We will see in Section 2.3 that, if a region of parameter space is unexplored, the value of σ for a candidate point chosen from that region will be very high. This algorithm therefore guards against the second shortcoming of MCMC (ignorance of disjoint regions of high likelihood) by explicitly stepping away from regions that are already known to be near the $(100 - \alpha)\%$ confidence limit and sampling points from poorly-sampled regions of parameter space. (Bryan 2007a) empirically compared this dependence on σ to other information-theoretic dependences and found it to perform best. Following that reference, we set $c = 1.96$ in equation (2).

Step (4A) chooses points solely based on their own merit, rather than the ratio of likelihoods used in steps (3M) and (4M) of MCMC. This algorithm can therefore ap-

ply a purely frequentist test to parameter space. Bounds are drawn in parameter space by examining the full set of points sampled and making a scatter-plot of those points which meet the $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$ criterion, rather than by integrating over the relative frequency with which the algorithm visited different points in parameter space. This guards against the final shortcoming of MCMC: the inherent subjectivity of Bayesian $(100 - \alpha)\%$ confidence limits.

2.3 Gaussian Processes

Step (3A) of APS generates a random set of candidate points and uses $\{\vec{\theta}_i, \mathcal{L}_i\}$ data from the points in parameter space already sampled to predict the value of \mathcal{L} at each candidate point (this prediction is μ in equation 2) and assign an uncertainty σ to that prediction. The algorithm is agnostic about how this prediction and uncertainty are derived. We follow (Bryan *et al.* 2007b) and use the formalism of Gaussian processes (more specifically: Kriging) to make the predictions. There is a slight difference between our formalism and theirs. This will be explained below. The following discussion comes mostly from Chapter 2 and Appendix A of (Rasmussen and Williams 2006).

Gaussian processes use the sampled data $\{\vec{\theta}_i, \mathcal{L}_i\}$ to predict \mathcal{L} at the candidate point $\vec{\theta}_q$ by assuming that the function $\mathcal{L}(\vec{\theta})$ represents a sample drawn from a random process distributed across parameter space. At each point in parameter space, \mathcal{L} is assumed to be distributed according to a normal distribution with mean $\bar{\mathcal{L}}$ and variance dictated by the covariance function $C_{ij}(\vec{\theta}_i, \vec{\theta}_j)$. C_{ii} is the intrinsic variance in the value of \mathcal{L} at a single point $\vec{\theta}_i$. C_{ij} encodes how variations in \mathcal{L} at one point in parameter space affect variations in \mathcal{L} at other points in parameter space. Rasmussen and Williams (2006) treat the special case $\bar{\mathcal{L}} = 0$ and finds (see their equations 2.19, 2.25 and 2.26)

$$\begin{aligned} \mu_q &= \sum_{i,j} C_{qi} C_{ij}^{-1} \mathcal{L}_j \\ \sigma_q^2 &= C_{qq} - \sum_{i,j} C_{qi} C_{ij}^{-1} C_{jq} \end{aligned} \quad (3)$$

where the sums over i and j are sums over the sampled points in parameter space. C_{iq} relates the sampled point i to the candidate point q . We do not wish to assume that the mean value of \mathcal{L} is zero everywhere. Therefore, we modify the equation for μ to give

$$\mu = \bar{\mathcal{L}} + \sum_{i,j} C_{qi} C_{ij}^{-1} (\mathcal{L}_j - \bar{\mathcal{L}}) \quad (4)$$

where $\bar{\mathcal{L}}$ is the algebraic mean of the sampled \mathcal{L}_i . Note the similarity to a multi-dimensional Taylor series expansion with the covariance matrix playing the role of the derivatives. Equation (3) differs from equation (6) in (Bryan *et al.* 2007b) because they used the semivariance

$$\gamma_{ij} = \text{var}[\mathcal{L}(\vec{\theta}_i) - \mathcal{L}(\vec{\theta}_j)]$$

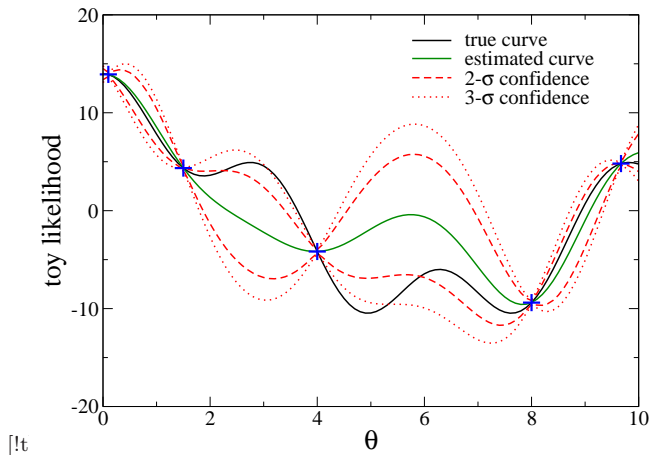


Figure 1. A one-dimensional example of prediction using Gaussian processes. The black curve is the function being considered. The crosses are the points at which it has been sampled. The green curve is the resulting prediction and the red curves represent the 2- and 3- σ uncertainty bounds. For the purposes of this illustration, we assumed that the Kriging parameter was $K = 10$.

in place of the covariance C_{ij} . In practice, the two assumptions result in equivalently valid μ and σ .

The form of the covariance function $C_{ij}(\vec{\theta}_i, \vec{\theta}_j)$ must be assumed. We choose to use

$$C_{ij} = K \exp \left[-\frac{1}{2} D_{ij}^2 \right] \quad (5)$$

where D_{ij} is the distance in parameter space between the points $\vec{\theta}_i$ and $\vec{\theta}_j$. The exponential form of C_{ij} quantifies the assumption that distant points should not be very correlated. The normalization constant K (known as the “Kriging parameter” for the geophysicist who pioneered the overall method) also must be assumed. This is somewhat problematic because, examining equation (4), one sees that the factors of K and K^{-1} completely factor out of the prediction μ , so that the assumed value of K has no effect on the accuracy of the prediction. If the opposite had been true, one could heuristically set K to maximize the accuracy of μ . Given that the function we are trying to model (\mathcal{L} as a function of set data and a specified theory) is not a random process, we find no a priori way to set K and instead set it according to heuristics that we believe are consistent with the behaviors we desire from the APS algorithm. We discuss this in more detail in Section 3.

Figure 1 applies the Gaussian process of equations (3) and (4) with assumption (5) to a toy one-dimensional function. Inspection shows many desirable behaviors in μ and σ . As θ_q approaches the sampled points θ_i , μ approaches \mathcal{L}_i and σ approaches zero. Closer to a sampled point, the Gaussian process knows more about the true behavior of the function. Far from the θ_i , σ is larger, and the S statistic in equation (2) will induce the APS algorithm to examine the true value of \mathcal{L} .

2.4 Comparison of Algorithms

Figure 2 directly compares APS with MCMC by running both on a toy model in two-dimensional parameter space. The model likelihood function is a χ^2 -distribution with two degrees of freedom. We are interested in the 95% confidence limit, which corresponds to a criterion of

$$\mathcal{L} \geq \mathcal{L}_{\text{lim}} \rightarrow \chi^2 \leq 6$$

The model is constructed so that there are two regions of parameter space that meet this criterion. These are the two ellipses in Figure 2(a). Figures 2(b) and 2(c) zoom in on one of these regions. The open circles correspond to the points sampled by MCMC. Each color represents an independent chain (four in all). Each of these chains was allowed to run until it had sampled 100 points in parameter space. The blue crosses represent the points sampled by APS after it had sampled a total of 400 points in parameter space.

Note that no single MCMC chain found both regions of high likelihood. A user would have to run multiple independent chains to be at all certain that she had discovered all of the regions of interest. It is now standard to address this by running several MCMC chains and aggregating their outputs, but it is concerning to leave coverage of the search space to the luck of multiple random restarts. Conversely, APS sampled points from distant regions of parameter space. This is how APS ultimately found both regions of high likelihood. Indeed, after only 85 calls to the likelihood function, APS had sampled points from both regions of high likelihood.

Consider Figures 2(c) and 2(b). It is obvious that, even near the most densely-sampled high-likelihood region, APS is principally interested in points on the boundary of the confidence region, while MCMC samples the interior. This is another way in which APS is a more efficient allocation of computing resources than MCMC.

Finally we can observe the perils of common MCMC convergence heuristics. They generally compare statistics from all the chains in aggregate against individual chains. At convergence, these statistics should be similar. We observe that in the toy example, convergence has not happened because some chains are near one local maximum while some are at the other. However, from a practical point of view all the high likelihood areas have been identified. Conversely, if the four chains had been chosen unluckily, they would have all converged to the same local maximum and never reached the other. The convergence tests would have reported success while the reality would have been failure to find both local maxima. We will present a more detailed comparison of MCMC and APS in Section 4.

3 TUNABLE PARAMETERS

APS as presented thus far contains parameters which must be set by the user. We describe them in this section. A summary list is presented at the end of the section.

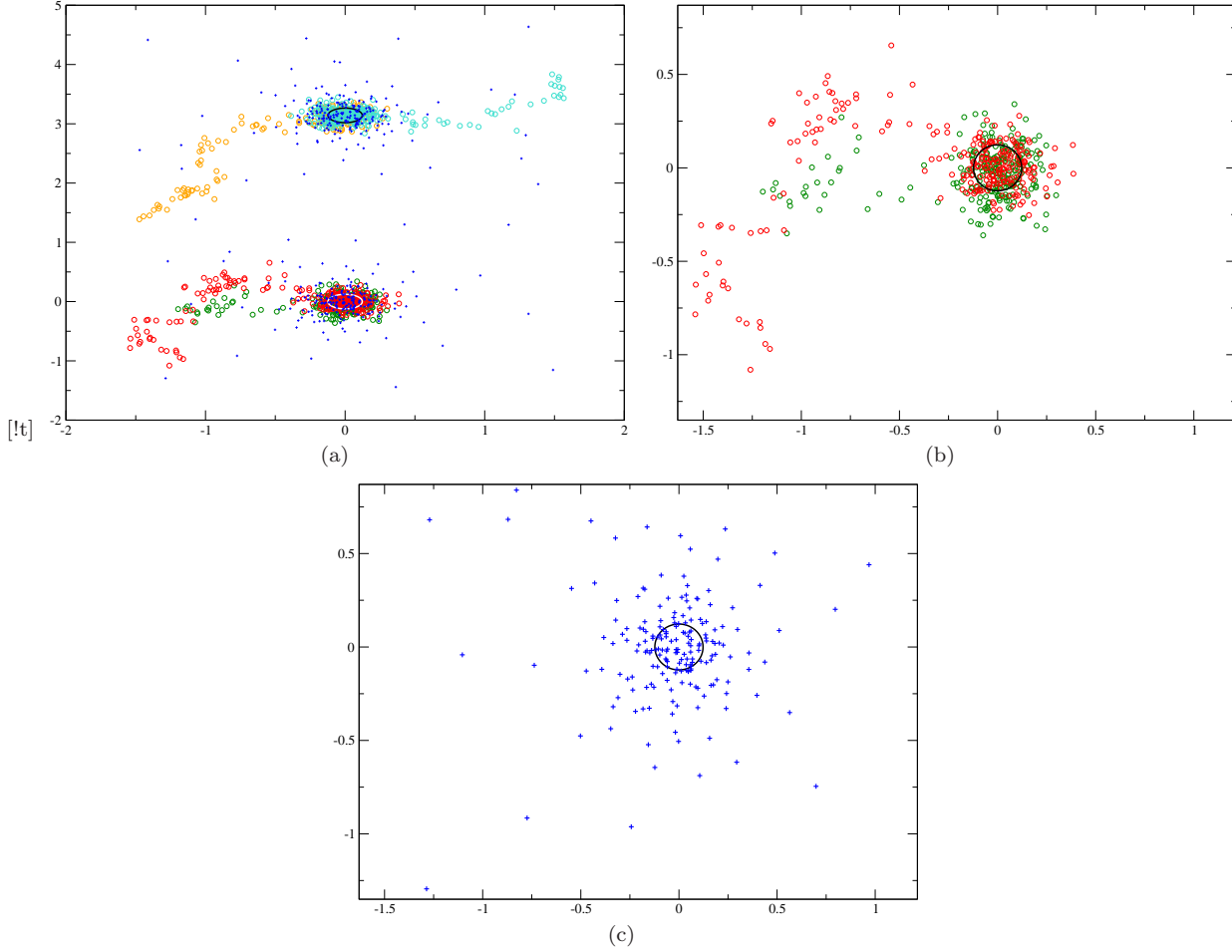


Figure 2. Points sampled by MCMC and APS on a toy likelihood function with two disjoint regions of high likelihood (the large ellipses) in a two-dimensional parameter space. Open circles are points sampled by MCMC. Each different color is an independent chain. The blue crosses are the points sampled by a single run of the APS algorithm run until it has sampled 400 points. Figures 2(b) and 2(c) zoom in on one of the high-likelihood regions to demonstrate that points sampled with the APS algorithm do not tend to cluster in the center of the high likelihood region as points sampled by MCMC tend to do. Also note that no single MCMC chain finds both regions of high likelihood. APS does find both.

N_C is the number of candidate points considered at each iteration. One consideration that should go into choosing N_C is speed. If N_C is too large, the evaluation of all N_C values of μ and σ in step (3A) will become comparably expensive to evaluating the likelihood function and the algorithm will lose some of its efficiency advantage over MCMC. N_C can also affect the algorithm's propensity to explore unknown regions of parameter space. A small N_C adds additional randomness because the selection will be affected more by the luck of choosing candidates than the metric used to evaluate them.

N_S is the number of purely random points on which to sample \mathcal{L} before proceeding to iterate steps (2A)-(4A). This number need only be sufficiently large to make the initial Gaussian process reasonable. It should not be so large as to be equivalent to a fine-gridding of parameter space,

which would defeat the purpose of having an efficient search algorithm.

In practice, one must specify bounds on each of the parameters beyond which it is not worth exploring. To prevent the relative sizes of each parameter's allowed range from affecting the performance of our Gaussian process, we amend the covariance function (5) to read

$$C_{ij}(\vec{\theta}_i, \vec{\theta}_j) = K \exp \left[-\frac{1}{2} \sum_a \left(\frac{\theta_{i,a} - \theta_{j,a}}{\theta_{\max,a} - \theta_{\min,a}} \right)^2 \right] \quad (6)$$

where i and j denote different points in parameter space and the sum over a is a sum over the dimensionality of the parameter space. Additionally, in order to prevent the Gaussian process from becoming prohibitively slow as more points are added to the set of sampled points, we follow

(Bryan *et al.* 2007b) and only use the N_G nearest sampled neighbors of each candidate point when predicting μ and σ . N_G is another choice made by the user.

To set the Kriging parameter K , we sample an additional uniform set of N_K points after the initial N_S sample but before proceeding to step (2A). For each of these points we both predict μ using the Gaussian process and sample the actual \mathcal{L} . We set K equal to the value necessary that 68% of these N_K points have $\mathcal{L} - \sigma < \mu < \mathcal{L} + \sigma$. As the algorithm runs, we periodically adjust K so that the search through parameter space strikes a balance between exploring unknown regions of parameter space and identifying regions of parameter space satisfying $\mathcal{L} \sim \mathcal{L}_{\text{lim}}$. Note that one can just as easily assume the value of K , this, however, runs the risk that the APS algorithm will either fail to explore outside of the discovered high-likelihood points (if K is too small and $\sigma \ll |\mu - \mathcal{L}_{\text{lim}}|$) or will ignore the high-likelihood region altogether (the opposite case).

We make one final modification to the APS algorithm as currently presented. Because of the absolute nature of Frequentist parameter constraints, a point in parameter space with likelihood $\mathcal{L} = 0.95 \times \mathcal{L}_{\text{lim}}$ is still not within the confidence limit. If the code finds such a point and immediately resumes its random search through the broader parameter space, it has not learned anything about the allowed values of $\vec{\theta}$. We therefore modify the code so that, whenever it finds a point with $\chi_{\text{lim}}^2 < \chi^2 \leq 1.1 \times \chi_{\text{lim}}^2$, it pauses in its search and spends some number N_L of evaluations trying to find a nearby point that is actually inside the confidence limit. To conduct this refined search, we borrow an idea from (Vanderplas and Connolly 2012) and note that equations (4) and (6) offer a straightforwardly differentiable function for μ in terms of our theoretical parameters. Much as equation (4) does a good job of characterizing the value of \mathcal{L} at an unknown point, the derivative of equation (4) should do a good job of characterizing the derivative of \mathcal{L} at a known point. This allows us to use gradient descent to walk from a point near the likelihood threshold towards a point that is inside the likelihood threshold. The method is as follows.

- (1g) Starting from the already sampled point $\vec{\theta}_q$ which is near the likelihood threshold, assemble a list of the N_G nearest neighbors from the set of sampled points, this time including $\vec{\theta}_q$ itself as the absolute nearest neighbor.

- (2g) Differentiate equation (4) to get

$$\frac{\partial \mu}{\partial \vec{\theta}} = \sum_{i,j}^{N_G} \frac{\partial C_{q,i}}{\partial \vec{\theta}} C_{i,j}^{-1} (\mathcal{L}_j - \bar{\mathcal{L}})$$

and differentiate equation (6) to get

$$\frac{\partial C_{q,i}}{\partial \theta_a} = \frac{\theta_{i,a} - \theta_{q,a}}{\theta_{\text{max},a} - \theta_{\text{min},a}} \times C_{q,i}$$

- (3g) Use this derivative to select a point in parameter space a small step away from $\vec{\theta}_a$ along the direction that will maximize the change in \mathcal{L} . Sample the likelihood function at this point. This point is now $\vec{\theta}_q$.

- (4g) Repeat steps (1g) through (3g) until you find some fiducial maximum likelihood \mathcal{L}_{max} or you have iterated N_L times, whichever comes first.

This modification to APS is referred to as the “lingering modification.”

To summarize, the APS parameters that must be tuned by the user are (values in parentheses are the values used on the 6 dimensional parameter space in Section 5)

- N_S (1000) – the number of random distributed starting points evaluated in step (1A) of APS .

- N_K (1000) – a number of randomly sampled points used to heuristically set the Kriging parameter K in equation (5).

- N_C (250) – the number of candidate points randomly generated in step (3A) of APS .

- N_G (15) – the number of nearest neighbors used in the Gaussian Process.

- N_L (100) – the maximum number of iterations to be spent using gradient descent in the lingering modification.

- χ_{lim}^2 , $\mathcal{L}_{\text{extlim}}$ ($\chi^2 = 1281$) – the confidence limit threshold value APS is trying to find.

- χ_{min}^2 , \mathcal{L}_{max} ($\chi^2 = 1197$) – the value used as a target by gradient descent in the lingering modification.

- The dimensionality of the parameter space and the maximum and minimum values each parameter is allowed to take.

4 A TOY COMPARISON

In this section, we will test APS ’s performance against that of MCMC on a toy likelihood function with two regions of high likelihood. The function exists in four dimensions. The high likelihood regions are centered on the points

$$\vec{\theta}_1 = \{-5.0, 0, 0, 0\} \quad \vec{\theta}_2 = \{5.0, 0, 0, 0\}$$

The function itself is characterized by a χ^2 statistic with 1199 degrees of freedom, which depends on $\vec{\theta}$ according to

$$\begin{aligned} \chi^2(\vec{\theta}) = & 1300.0 + \frac{d_1^2}{2} + \frac{d_2^2}{2} \\ & + 1147 \exp[-d_1^2] + 1200 \exp[-0.5d_2^2] \end{aligned} \quad (7)$$

where $d_{1,2}$ are the distances in parameter space from $\vec{\theta}_{1,2}$. In this case, the 95% confidence limit threshold corresponds

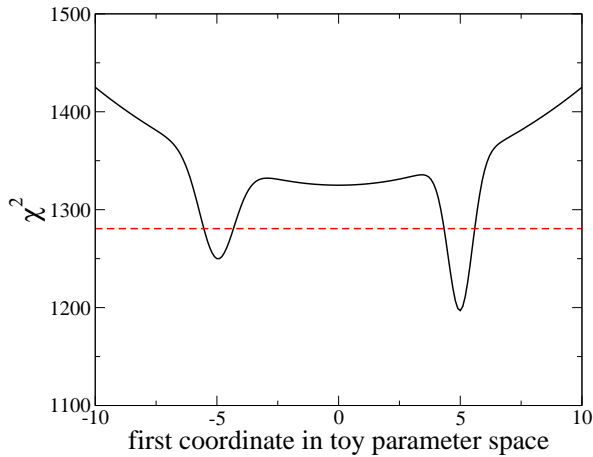


Figure 3. A one dimensional slice of our toy likelihood function. The red dashed line corresponds to the 95% confidence limit threshold for a χ^2 distribution with 1199 degrees of freedom.

to a value of $\chi^2 = 1281$. Figure 3 shows a one-dimensional slice of this function. The red dashed line corresponds to the threshold χ^2 limit. When testing APS, we set the tunable parameters thus

$$\begin{aligned} N_S &= 20 \\ N_K &= 20 \\ N_G &= 15 \\ N_C &= 250 \\ N_L &= 50 \\ \chi_{\min}^2 &= 1197 \end{aligned}$$

Before running either MCMC or APS, we generate three test grids each of 10,000 known points on this likelihood surface. We will use these grids to measure how long it takes MCMC and APS to learn the entire behavior of this likelihood surface. The first test grid is uniformly distributed across the entire likelihood surface (all θ_a parameters are allowed to vary from -10 to 10). The second grid is spherically distributed about the rightmost high likelihood region in Figure 3. The third grid is spherically distributed about the leftmost high likelihood region. The first grid contains no points whose χ^2 values are below the threshold limit. The second and third grids are roughly half comprised of points that are below the threshold.

To compare the efficacy of APS and MCMC at characterizing the likelihood surface, we run each algorithm 200 times. In the case of MCMC an individual “run” consists of four independent chains run in parallel. During each run, we periodically stop and consider the points sampled by each algorithm thus far. We treat these points as the input to a Gaussian process which we use to guess the χ^2 values of the points on our three test grids. We quantify the efficacy of the

algorithms in terms of the number of mischaracterizations made on each grid. We define a “mischaracterization” as a point on the test grid which satisfies $\chi^2 \leq \chi_{\text{lim}}^2$ but for which the Gaussian process predicts $\chi^2 > \chi_{\text{lim}}^2$ or vice-versa. Figure 4 shows the performance of the algorithms on this test averaged over all the 200 instantiations of each algorithm. You will note that while both algorithms were essentially perfect at characterizing the widely distributed test grid 1 (on which no points satisfied $\chi^2 \leq \chi_{\text{lim}}^2$; note the logarithmic vertical axis in Figure 4(a)), only APS with lingering successfully and reliably characterized both test grids centered on the high-likelihood regions in Figure 3. This is an illustration of the second shortcoming of MCMC identified in Section 2.1: once an MCMC chain has identified a high likelihood region, it is unlikely to step out of that region and consider the possibility that other high likelihood regions exist. This problem persists even though each MCMC instantiation consists of four independent chains, each with its own opportunity to fall into one or the other high likelihood region. Either the chains all fell into one high likelihood region and not the other, or the chains became trapped at the local minimum in equation (7) at $\bar{\theta} = \{0, 0, 0, 0\}$. In Section 5 we perform a similar test using actual data from the WMAP CMB experiment (Jarosik *et al.* 2011; WMAP 2010) and the 6-dimensional parameter space of the flat concordance cosmology.

5 TEST ON WMAP 7 YEAR DATA

In this section, we test the APS algorithm on the 7-year data release of the WMAP satellite, which measured the temperature and polarization anisotropy spectrum in the cosmic microwave background (Jarosik *et al.* 2011). For simplicity, we only consider anisotropies in the temperature-temperature correlation function and modify the likelihood function (WMAP 2010) to work in the space of anisotropy C_ℓ s, rather than working directly in the pixel space. This results in a likelihood function sampled from a χ^2 -distribution with 1199 degrees of freedom. In this case, the $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$ criterion for the 95% confidence limit corresponds to $\chi^2 \leq \chi_{\text{lim}}^2$ with $\chi_{\text{lim}}^2 = 1281$.

We take as our parameter space the six dimensional parameter space describing the set of spatially flat Λ CDM (cosmological constant and cold dark matter) concordance cosmologies. Those parameters are $\{\Omega_b h^2, \Omega_{\text{CDM}} h^2, h, \tau, n_s, \ln[10^{10} A]\}$. These parameters will be familiar to users of the MCMC code CosmoMC (Lewis and Bridle 2002) as the relative density of baryons, the relative density of dark matter, the present day Hubble parameter, the optical depth to last scattering, the spectral index controlling the scale-dependence of primordial density fluctuations in the universe, and a normalization parameter controlling the amplitude of primordial density fluctuations in the universe. We use the Boltzmann code

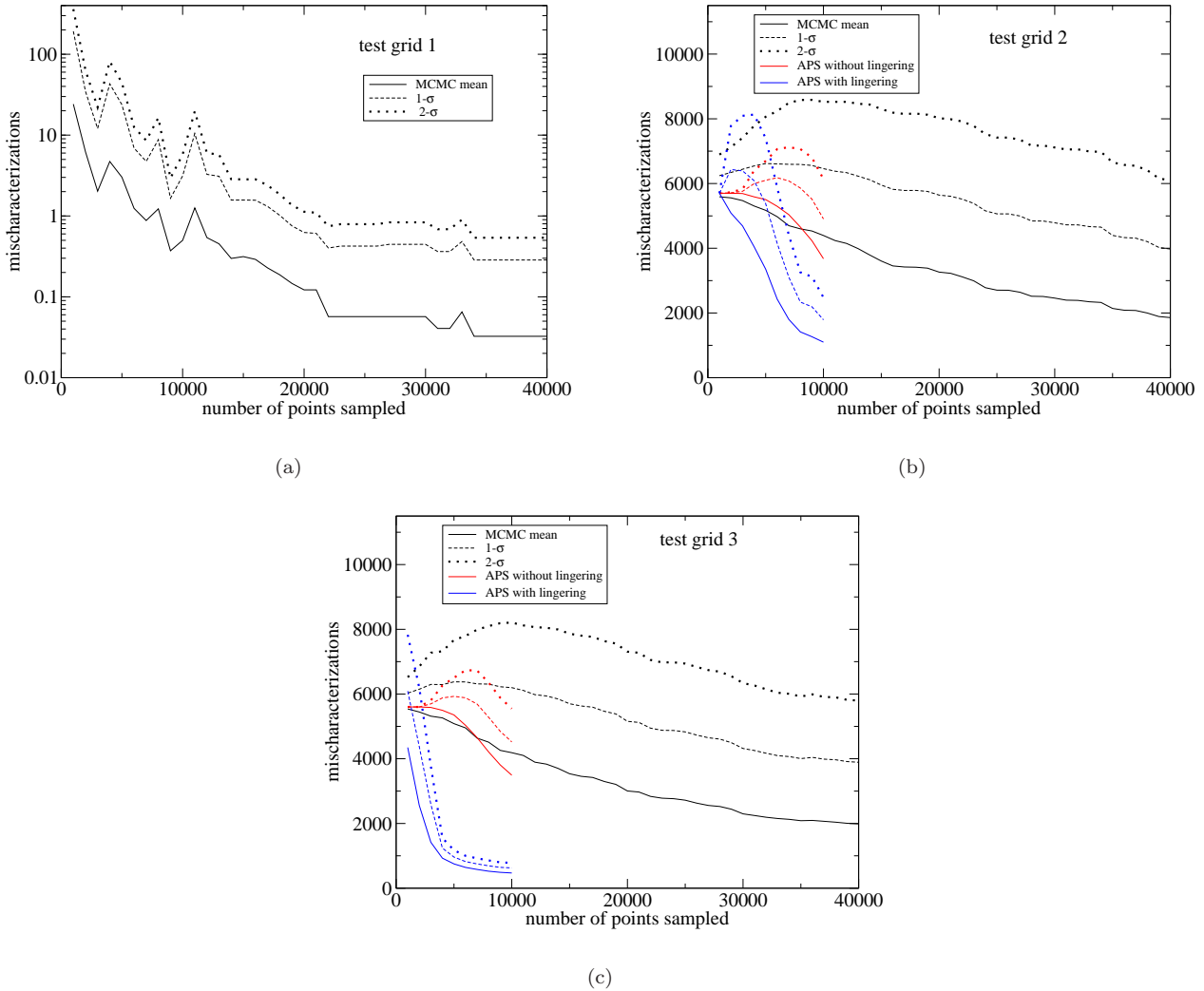


Figure 4. The performance of MCMC and APS at characterizing the points on the test grids built around the likelihood function in Figure 3. Solid lines are the mean number of mischaracterizations averaged over 200 instantiations of the algorithm. Dashed lines are the mean plus one standard deviation. Dotted lines are the mean plus two standard deviations. Black lines are MCMC. Red lines are APS without the lingering modification. Blue lines are APS with the lingering modification. The APS lines do not appear in Figure 4(a) because neither version of APS resulted in any mischaracterizations. Recall that each test grid contained 10,000 points.

CAMB to convert from parameter values to anisotropy spectra (Lewis, Challinor, and Lasenby 2000).

In this case, we test APS with

$$\begin{aligned}
 N_S &= 1000 \\
 N_K &= 1000 \\
 N_C &= 250 \\
 N_G &= 15 \\
 N_L &= 100 \\
 \chi^2_{\text{lim}} &= 1281 \\
 \chi^2_{\text{min}} &= 1197
 \end{aligned}$$

Our comparison MCMC is run by the publically available software CosmoMC (Lewis and Bridle 2002). We compare the results of the two algorithms – both in terms of derived constraints on the cosmological parameters and in terms of exploration of the full parameter space – below.

5.1 Parameter Constraints

As discussed in Section 2, MCMC determines parameter constraints by integrating over the posterior probability distribution on parameter space while the APS algorithm determines constraints by listing found points which satisfy

$\mathcal{L} \geq \mathcal{L}_{\text{lim}}$ and determining the region of parameter space spanned by those points. Figure 5 compares these two approaches by tracking the development of the 95% confidence limit contour in one two-dimensional slice of our full six-dimensional parameter space as a function of the number of points sampled by each algorithm. In each frame, the solid contours represent contours drawn by considering all of the points found by each algorithm which satisfy the Frequentist $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$ requirement. The blue contour represents points found by APS. The red contour represents points found by MCMC. The black contour represents points found by MCMC after sampling a total of 1.2 million points. Note: for this comparison, we consider all of the points visited by MCMC, not just those points accepted in steps (3M) and (4M) from Section 2.1. This is, in some sense, a more complete set of information about the likelihood surface than MCMC usually returns. The dashed black contour is the contour drawn using the usual Bayesian inference on the points accepted by MCMC after the full 1.2 million points have been sampled. The green crosses are the pixels in this 2-dimensional parameter space that Bayesian inference believes are inside of the 95% confidence limit after the specified number of points have been sampled.

The salient features of this figure are as follows. The solid black contour is larger than the dashed black contour. This means that MCMC visited points that met the Frequentist threshold $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$ but not with enough frequency to satisfy the 95% Bayesian confidence limit. This is an example of MCMC being too restrictive as discussed in Section 2.1. A similar conclusion can be drawn from the fact that the green crosses do not fill the red contour until 400,000 points have been sampled.

The green crosses congregate in the center of the contours because MCMC is principally interested in the deep interior of the high likelihood region. This is a manifestation of the inefficiency of MCMC discussed in Section 2.1.

The blue and red contours track quite well at all points in the algorithms' histories. This shows that APS is at least as good as MCMC at deriving parameter constraints when you treat the points visited by MCMC from a Frequentist perspective. Comparing the blue contour to the green crosses, in Figure 5(c), one sees that APS derives accurate parameter constraints faster than MCMC treated from a Bayesian perspective. Figure 6 recreates Figure 5(d), except that the green crosses represent the 50% confidence limit according to Bayesian MCMC after sampling 400,000 points. The fact that these pixels still occur outside of the solid black contour (Frequentist MCMC 95% confidence limit after sampling 1.2 million points) indicates that the false positives in Figure 5(d) represent a significant fraction of the total posterior probability integrated by Bayesian MCMC at this point in the algorithm. In contrast, the APS contour already covers 86% of the final area of the Frequentist MCMC contour. Because of how they are drawn ($\mathcal{L} \geq \mathcal{L}_{\text{lim}}$), APS contours will not include false positives.

Figure 7 plots the one-dimensional 95% confidence lim-

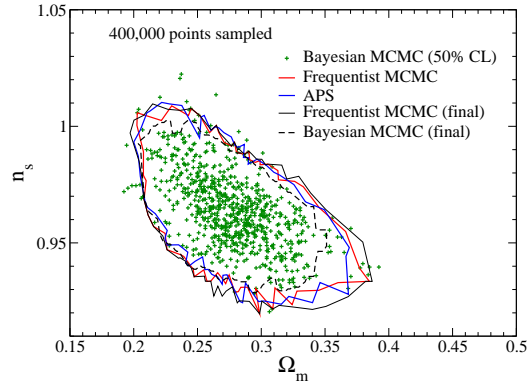


Figure 6. This figure recreates Figure 5(d) except that now the green crosses are the 50% Bayesian MCMC confidence limit. The fact that these crosses still occur outside of the solid black contours indicates that false positives account for a large fraction of the total posterior probability integrated by MCMC, even after sampling 400,000 points in parameter space.

its on our six cosmological parameters. Again, the solid red lines consider all of the points visited by MCMC (not just those accepted by the chain) and set the limit according to the Frequentist threshold. The blue lines represent the results from APS. Note that APS only sampled 420,000 points before we stopped it. MCMC sampled 1.2 million points. The dashed black lines are the confidence limits inferred from Bayesian analysis on only those points accepted by the MCMC chains. As in Figure 5, we see that APS converges to the same answers as Frequentist MCMC in comparable time, and that both Frequentist analyses find allowed parameter values that are missed by the Bayesian analysis. Cases where the dashed black lines stray outside of the solid red lines indicate Bayesian analysis applying spurious weight to low-likelihood points.

At this point, we have made the case that APS gives more accurate parameter constraints in less time than the usual, Bayesian MCMC analysis. However, even if one were simply to modify their MCMC analyses to adopt a Frequentist perspective (the red contours and lines in Figures 5 and 7), we show in Section 5.2 below that APS exhibits superior performance characterizing the entire likelihood surface, not just the high likelihood subsurface. In the language of Section 4, APS gives constraints as accurate as MCMC with greater confidence that you have not ignored any additional regions of high likelihood.

5.2 Exploration of Parameter Space

Section 5.1 examined the performance of the APS algorithm within the high likelihood region of parameter space by comparing the derived parameter constraints to those found by MCMC. This section will consider the performance of the APS algorithm in the low likelihood region of parameter

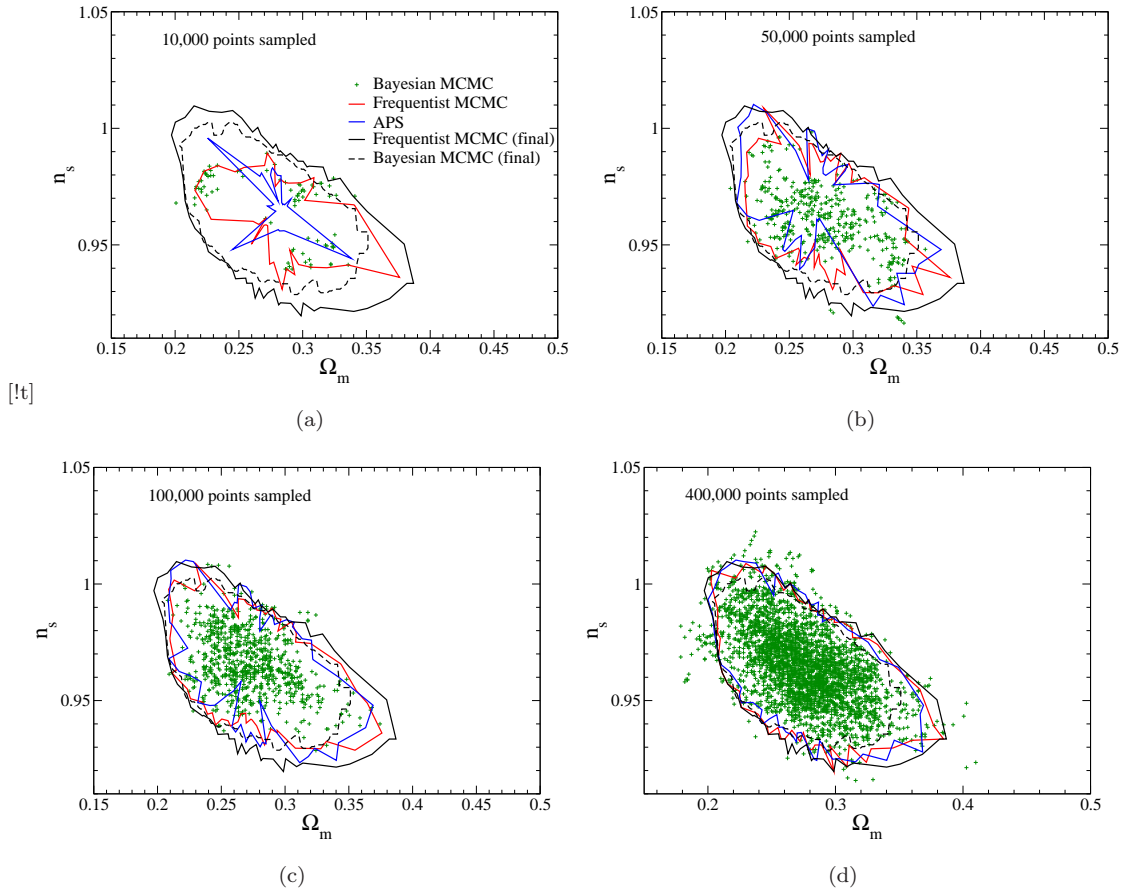


Figure 5. The 95% confidence limit contours in a two-dimensional slice of the full six-dimensional cosmological parameter space as determined by APS and MCMC at different stages in the algorithm’s progression. The solid curves represent contours drawn by considering all of the $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$ points found by each algorithm. The solid red curve is MCMC after sampling the specified number of points. The solid blue curve is APS after sampling the specified number of points. The solid black curve is MCMC after sampling 1.2 million points. This will be taken to be the truest answer. The dashed black curve represents the contour drawn using the usual Bayesian inference on the full 1.2 million-point MCMC chain. The green crosses represent the pixels that Bayesian inference declares to be inside of the confidence interval after MCMC has sampled the number of points specified.

space, asking the question “how certain can we be that the excluded regions of parameter space contain no points of interest?” Recall the toy model in Section 4: MCMC is notoriously inefficient (or even ineffective) at exploring multi-modal likelihood functions. By selecting sample points based both on proximity to the confidence limit and on uncertainty in the Gaussian process prediction σ , the APS algorithm ought to improve on that performance.

To test this hypothesis, we perform the test illustrated in Figure 4 on the WMAP parameter space and likelihood surface. We generate two test grids each of 1,000,000 points. One grid is distributed uniformly across parameter space. This grid contains no good points that satisfy the $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$ criterion. The other grid is spherically distributed about the vicinity of the high likelihood region and contains 110,000 good points. We then take the points sampled by

MCMC and APS (again, we take all of the points sampled by MCMC; not just those points accepted in the chains) at different times in their history and use those points as the input for a Gaussian process, which we use to predict the \mathcal{L} values of the points on our test grid. Figure 8 shows the number of points that are thus misclassified ($\mathcal{L} \geq \mathcal{L}_{\text{lim}}$ points for which the Gaussian process predicts $\mathcal{L} < \mathcal{L}_{\text{lim}}$ and vice-versa) as a function of the number of points fed into the Gaussian process. Because the WMAP 7 likelihood function is so expensive, we only ran this test once. To find the confidence limit (dashed) curves, we consider the uncertainty implied by the Gaussian process σ in equation (3). The dashed curves encompass points that are within σ of being misclassified (i.e. points for which $\chi^2 \leq \chi_{\text{lim}}^2$ but $\mu + \sigma > \chi_{\text{lim}}^2$ and vice versa).

As you can see from Figure 8(a), APS does a much

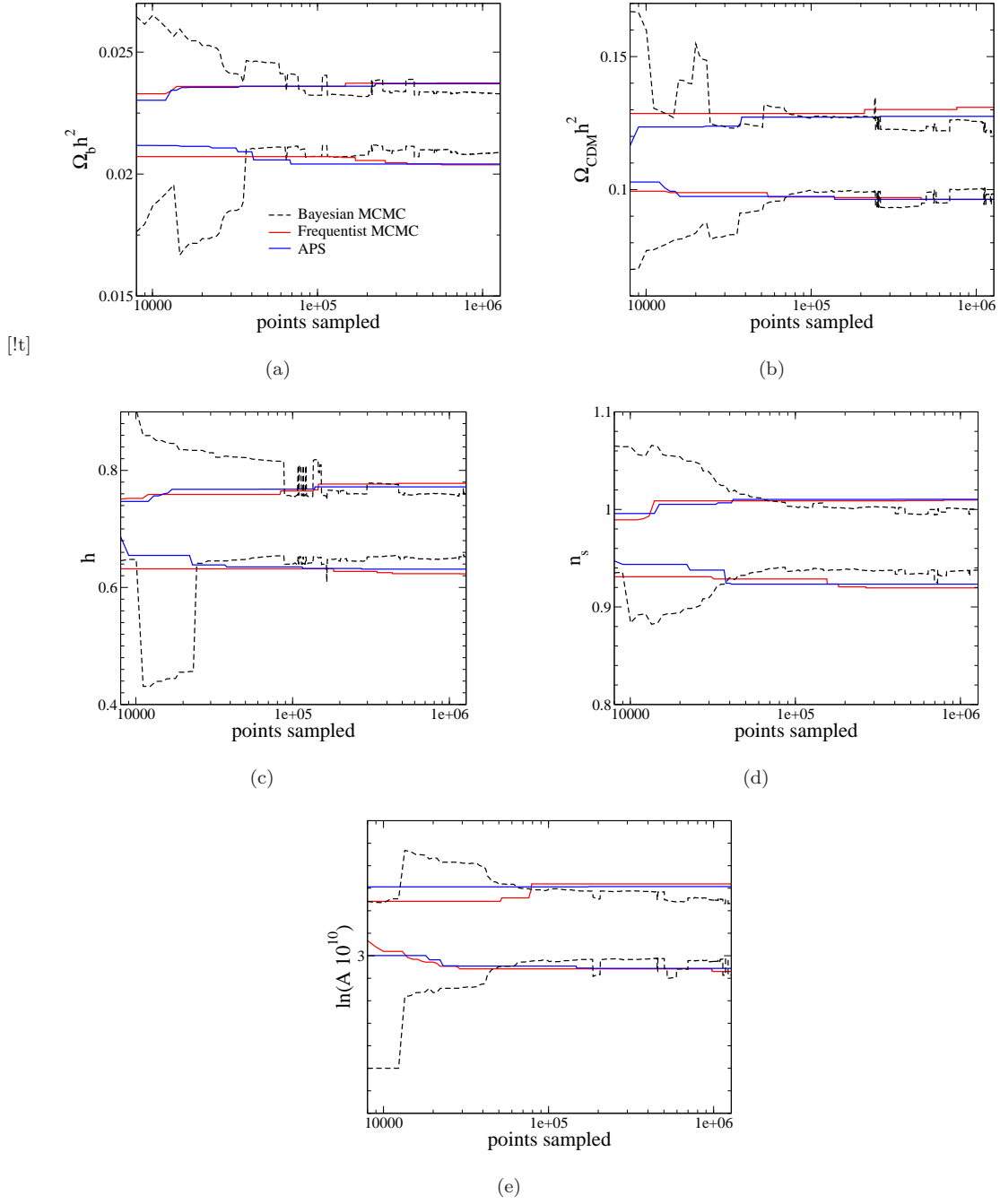


Figure 7. These figures plot the 95 % confidence limit bounds on our cosmological parameter set as derived by MCMC and APS as a function of the number of points sampled. The red solid line treats the output from MCMC using a Frequentist perspective. The blue solid line treats the output from APS (also as a Frequentist). The dashed black line treats the output from MCMC using the usual Bayesian perspective.

better job at characterizing the uniform test grid than does MCMC. Figure 8(b) shows that the two algorithms perform comparably poor on the high likelihood test grid. This is likely due to the compact nature of the high likelihood region of the WMAP 7 likelihood surface. It is small both in terms of extent on parameter space and in terms of the difference in χ^2 between likely and unlikely points. Recall that the 95% confidence limit we are considering corresponds to $\chi^2 = 1281$. The smallest χ^2 found by either algorithm was $\chi^2 \sim 1270$. This small difference between likely and unlikely points means that even a fraction of a percent error in the value of χ^2 predicted by our test grid Gaussian process will result in a mischaracterization. Figure 9 shows that this is indeed what happens. Here we plot the fractional error in predicted χ^2 as a function of actual χ^2 for both algorithms. Red curves are MCMC. Blue curves are APS. Dotted curves are results after sampling 50,000 points. Dashed curves are results after sampling 100,000 points. Solid curves are results after sampling 400,000 points. As you can see from Figure 9(a), there is indeed imprecision on the order of 1% when considering test points near the 95% confidence limit threshold. On a more forgiving likelihood surface with greater $\Delta\chi^2$ between likely and unlikely points, this would not result in the large number of mischaracterizations evident in Figure 8(b). The WMAP 7 likelihood surface is anything but forgiving.

Readers interested in seeing how APS learns about the likelihood surface over time can consider Figures 9(b), which recreates Figure 9(a) for a broader expanse of χ^2 , and Figure 10, which shows the fraction of mischaracterized points on both test grids as a function of χ^2 values. As you can see, while APS learns rapidly about the unlikely regions of parameter space, MCMC remains largely oblivious to what is going on in the regions outside of its integral bounds.

These results, combined with the parameter constraints illustrated in Section 5.1 cause us to conclude that APS is at least as effective as MCMC and locating regions of high likelihood parameter space, and is significantly more robust against anomalies in regions of low likelihood parameter space.

6 DISCUSSION

APS as demonstrated here has three principal advantages over MCMC when deriving parameter constraints.

- (ia) APS is more computationally efficient than MCMC in that it does not spend time exploring the interior of high likelihood regions when only their bounds are of interest. As a result it can yield comparable parameter constraints with significantly fewer calls made to expensive likelihood functions.

- (iia) APS allows for simultaneous robust statements about both high and low likelihood regions of parameter

space. MCMC is robust only in the high likelihood region it happens to discover.

- (iiia) APS allows investigators to apply frequentist assumptions to their parameter constraints.

The shortcomings of APS are twofold.

- (ib) APS has no framework for exploring a likelihood function whose form is not known (this is the corollary to (iia) above). You must specify \mathcal{L}_{lim} or χ^2_{lim} before running APS. You cannot use APS to discover \mathcal{L}_{lim} .

- (iib) There is no well-accepted stopping criterion for the algorithm equivalent to the convergence criteria usually applied to MCMC (for example, (Brooks and Gelman 1998)). However, as we observed on the toy problem and as Bryan et al. observed by finding a second high likelihood area in the WMAP data, MCMC's convergence provides a false sense of security.

- (iiib) APS is much more complicated to implement than the most basic MCMC. We hope that by making our code publically available, we can help the community to overcome this hurdle.

APS may be used as a more efficient alternative to MCMC. They may also be combined. Often, the convergence of MCMC chains is dependent on the size of parameter space to be explored. The larger the region, the slower convergence. Investigators can exploit advantage (ia) by pre-processing their data with APS and using the discovered high likelihood regions to set the prior bounds for their MCMC analyses.

We make our code available at <https://github.com/uwssg/APS>. The code is presented as a series of C++ modules with directions indicating where the user can interface with her particular likelihood function. Those with questions about the code or the algorithm should not hesitate to contact the authors.

ACKNOWLEDGMENTS

SFD would like to thank Eric Linder, Arman Shafieloo, and Jacob Vanderplas for useful conversations about Gaussian processes. SFD would also like to acknowledge the hospitality of the Institute for the Early Universe at Ewha Womans University in Seoul, Korea, who hosted him while some of this work was done. We acknowledge support from DOE award grant number DESC0002607 and NSF grant IIS-0844580.

REFERENCES

Atchadé, Y. F. and Rosenthal, J. S., 2005, *Bernoulli* **11**, 815

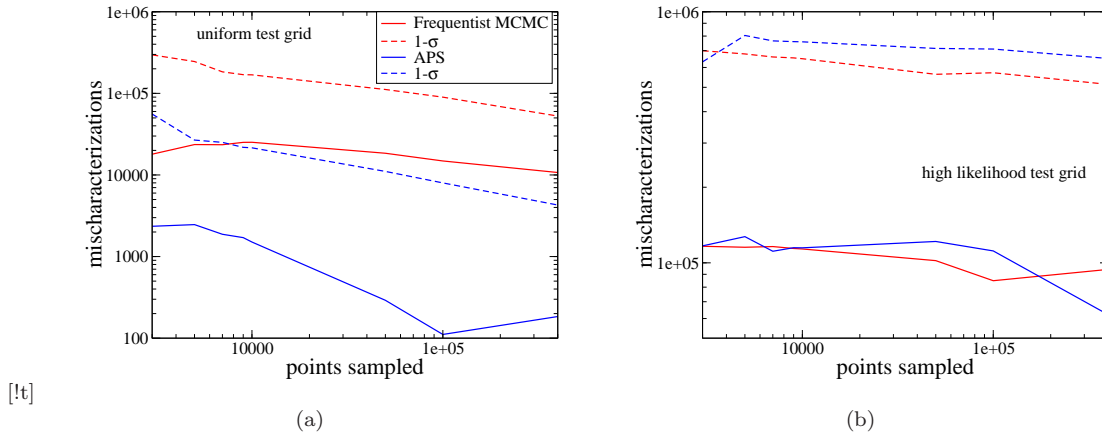


Figure 8. The same test we performed on our toy model in Figure 4 this time using the WMAP7 parameters and likelihood function. The “uniform test grid” is distributed uniformly across parameter space and contains no good points that satisfy $\mathcal{L} \geq \mathcal{L}_{\text{lim}}$. The “high likelihood test grid” is spherically distributed about the high likelihood region and contains 110,000 good points. Red curves consider points sampled by MCMC. Blue curves consider points sampled by APS. See the text for explanation of dashed curves. The vertical axes count the number of points that are misclassified by the test Gaussian process.

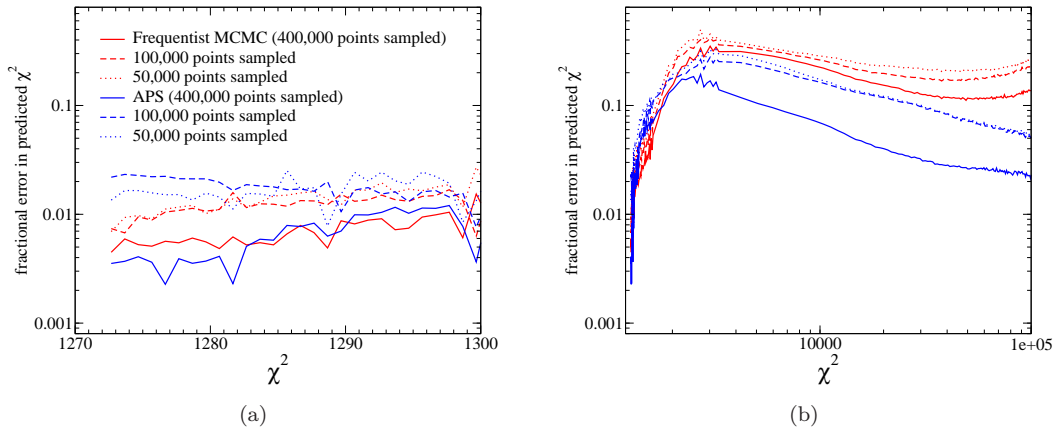


Figure 9. The average fractional error in the χ^2 predicted by the test Gaussian process used to generate Figure 8. The horizontal axis is the test points’ actual value of χ^2 . Red curves are MCMC. Blue curves are APS. Solid curves are results after 400,000 points have been sampled. Dashed curves are results after 100,000 points have been sampled. Dotted curves are results after 50,000 points have been sampled.

Brooks, S. and Gelman, A., 1998, *Journal of Computational and Graphical Statistics* **7** 434
 Bryan, B., 2007a, Ph.D. thesis <http://reports-archive.adm.cs.cmu.edu/anon/ml2007/abstracts/07-122.html>
 Bryan, B., Schneider, J., Miller, C. J., Nichol, R. C., Genovese, C., and Wasserman, L., 2007b, *Astrophys. J.* **665**, 25
 Dunkley, J. Bucher, M., Ferreira, P. G., Moodley, K., and Skordis, C., 2005, *Mon. Not. R. Astron. Soc.* **356**, 925
 Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. ed., “Markov Chain Monte Carlo in Practice,” (Chapman & Hall, 1996), Boca Raton, F.L.
 N. Jarosik *et al.*, 2011, *Astrophys. J. Suppl. Ser.* **192** 14

Lewis, A. and Bridle, S., 2002, *Phys. Rev. D* **66**, 103511; <http://cosmologist.info/notes/COSMOMC.ps.gz>
 Lewis, A., Challinor, A., and Lasenby, A., 2000, *Astrophys. J.* **538**, 473
 Rasmussen, C. E. and Williams, C. K. I., 2006, “Processes for Machine Learning” <http://www.GaussianProcess.org/gpml>
 Vanderplas, J. and Connolly, A., 2012, [in preparation]
 WMAP, 2010 http://lambda.gsfc.nasa.gov/product/map/current/likelihood_info.cfm

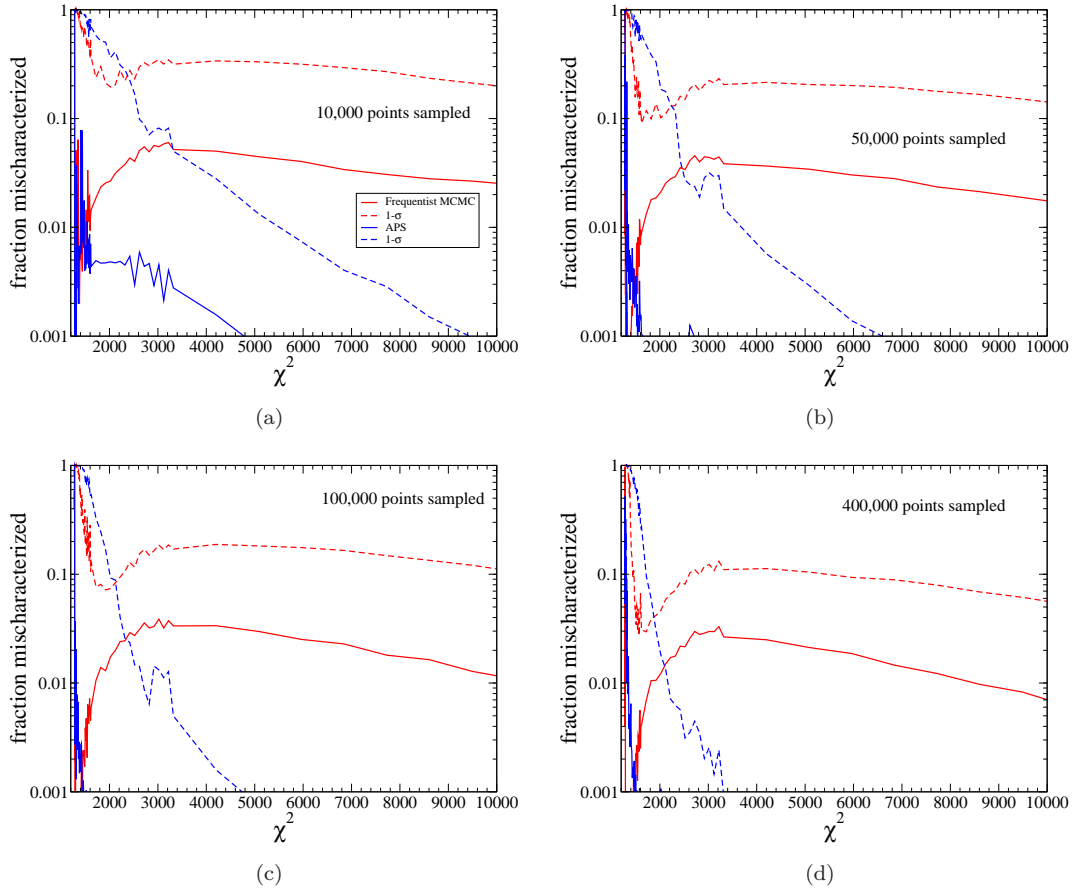


Figure 10. These plots combine both of the test grids from Figure 8 and show the fraction of points misclassified by the test Gaussian process as a function of their true χ^2 value. Red curves consider points sampled by MCMC. Blue curves consider points sampled by APS. Dashed curves are as in Figure 8.